Module 8: Hands-On: Basic Terraform Operations

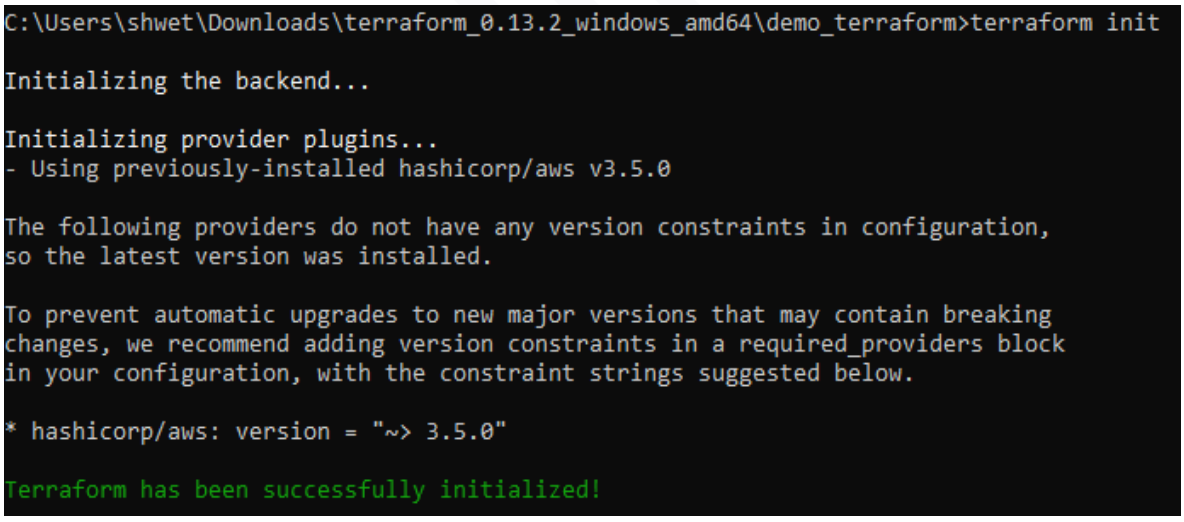
Basic Terraform Operations:

1. **terraform init:** The terraform init command is used to initialize a working directory containing Terraform configuration files. Open a text editor and type:

```
provider "cloud Provider"{
region= "any region"
access_key="Your access Key"
secret_key="Your Secret Key"
}
```

Save this file inside the same directory of the Terraform. Now open the Command prompt, navigate to your project folder and type terraform init. It will initialize all the provider's plugins.

```
terraform init
```



2. **terraform plan:** The terraform plan command is used to create an execution plan. Now we have the code written in the text editor and we will tell Terraform to plan it. This will create an AWS instance with the specified AMI and the type.

```
provider "aws"{
region= "us-west-2"
access_key = “your Access Key"
secret_key = “Your secret key”
}
resource "aws_instance" "example" {
        ami="ami-0ba60995c1589da9d"
        instance_type="t2.micro"
```

This command is a convenient way to check whether the execution plan for a set of changes matches your expectations without making any changes to real resources or to the state.

```
terraform plan
```

```
C:\Users\shwet\Downloads\terraform_0.13.2_windows_amd64\demo_terraform>terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.


------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create
```


3. **terraform apply:** The terraform apply command is used to apply the changes required to reach the desired state of the configuration, or the predetermined set of actions generated by a terraform plan execution plan. After terraform apply, an AWS instance will be created on the EC2.

```
terraform apply
```

```
Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.example: Creating...
aws_instance.example: Still creating... [10s elapsed]
aws_instance.example: Still creating... [20s elapsed]
aws_instance.example: Still creating... [30s elapsed]
aws_instance.example: Still creating... [40s elapsed]
aws_instance.example: Creation complete after 45s [id=i-0044a4ddba25d5c47]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

4. **terraform show:** The terraform show command is used to see all the configuration and to ensure that the planned operations are expected.

```
terraform show
```



```
C:\Users\shwet\Downloads\terraform_0.13.2_windows_amd64\demo_terraform>terraform show
# aws_instance.example:
resource "aws_instance" "example" {
    ami                          = "ami-0ba60995c1589da9d"
    arn                          = "arn:aws:ec2:us-west-2:697042039271:instance/i-0044a4ddba25d5c47"
    associate_public_ip_address  = true
    availability_zone            = "us-west-2c"
    cpu_core_count               = 1
    cpu_threads_per_core         = 1
    disable_api_termination      = false
    ebs_optimized                = false
    get_password_data            = false
    hibernation                  = false
    id                           = "i-0044a4ddba25d5c47"
    instance_state               = "running"
    instance_type                = "t2.micro"
    ipv6_address_count           = 0
    ipv6_addresses               = []
    monitoring                   = false
    primary_network_interface_id = "eni-0480241643bca3885"
    private_dns                  = "ip-172-31-13-59.us-west-2.compute.internal"
    private_ip                   = "172.31.13.59"
    public_dns                   = "ec2-52-12-148-66.us-west-2.compute.amazonaws.com"
    public_ip                    = "52.12.148.66"
    secondary_private_ips        = []
    security_groups              = [
        "default",
```

5. **terraform validate:** The terraform validate command is used to validate the configuration and checks whether a configuration is syntactically valid. It is used for general verification of correctness of attribute names and value types.

```
terraform validate
```

```
C:\Users\shwet\Downloads\terraform_0.13.2_windows_amd64\demo_terraform>terraform validate
Success! The configuration is valid.


C:\Users\shwet\Downloads\terraform_0.13.2_windows_amd64\demo_terraform>
```

